**TECHNICAL UNIVERSITY**



**Faculty of Automation and Computer Science**

**Pop Tudor**

**Petrean Florin**

**Group 30432**

**Teaching Assistant : Ion Augustin Giosan**

**Contents**

1.Introduction

2.Theoretical background

3.Design and Implementation

4.Experimental Results

5.Conclusions

6.Bibliography

**1.Introduction**

The aim for this project is design and implement using OpenCV library a program to detect road signs from various images. This program will be helpful in the future, solving the problem of road signs detection for self-driving cars.

**2.Theoretical Background**

For the image processing part of the project, a good approach to detecting road signs is to have an already established database of those and compare them with the fetched road sign from our source image and see if they match at least 90%.

As the road signs are universal in color, we can fetch those respective colors out of the image. Doing this, we have to take into consideration that there could be different shades of the color (because of shadows, light, etc.) in the source image which could not be detected. In order to bypass this problem, the source image can be transformed from RGB space to HSV space so the pure color(H) can be detected. The H value from the database image will be compared with the H value of the source image. The S and V components of the image can also be used in some way to better approximate the road sign. We can also improve the detection by selecting a range of colors(a threshold) that are treated as valid.

Lastly, we need some sort of edge detection (that means detecting when there is a big color change from one pixel to another) to see where the road sign edges begin and end. After this detection has taken place, we need to compare the edges of each object and see if it matches at least 90%.

**3.Design and implementation**

So the basic algorithm would be:

1. Transfer the source image to HSV

* using the method given at the laboratory session

1. Compare the colors with the predefined road sign colors and adjust with a threshold.

* iterate through each road sign colour and try to find it in the source image

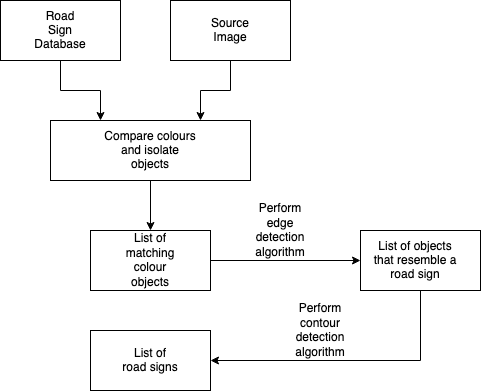
1. Fetch the road sign from the source image

* analyze and detect all objects that have colour matching with any of the colours from our road sign database(array of images)
* store all objects in their respective colour category

1. Compare the edges with the database images.

* use an edge detection algorithm and compare the edges of each object with the road sign database shapes
* use an contour detection algorithm to verify the number of edges given by the edge detection algorithm

**Flow-chart Diagram**



**Black Box view of the project:**

****

**4.Experimental Results**

**5.Conclusions**

**6.Bibliography**